

**No.7**

# **Generative Models: Parameter Estimation**

*Prof. Hui Jiang*

Department of Computer Science and Engineering  
York University

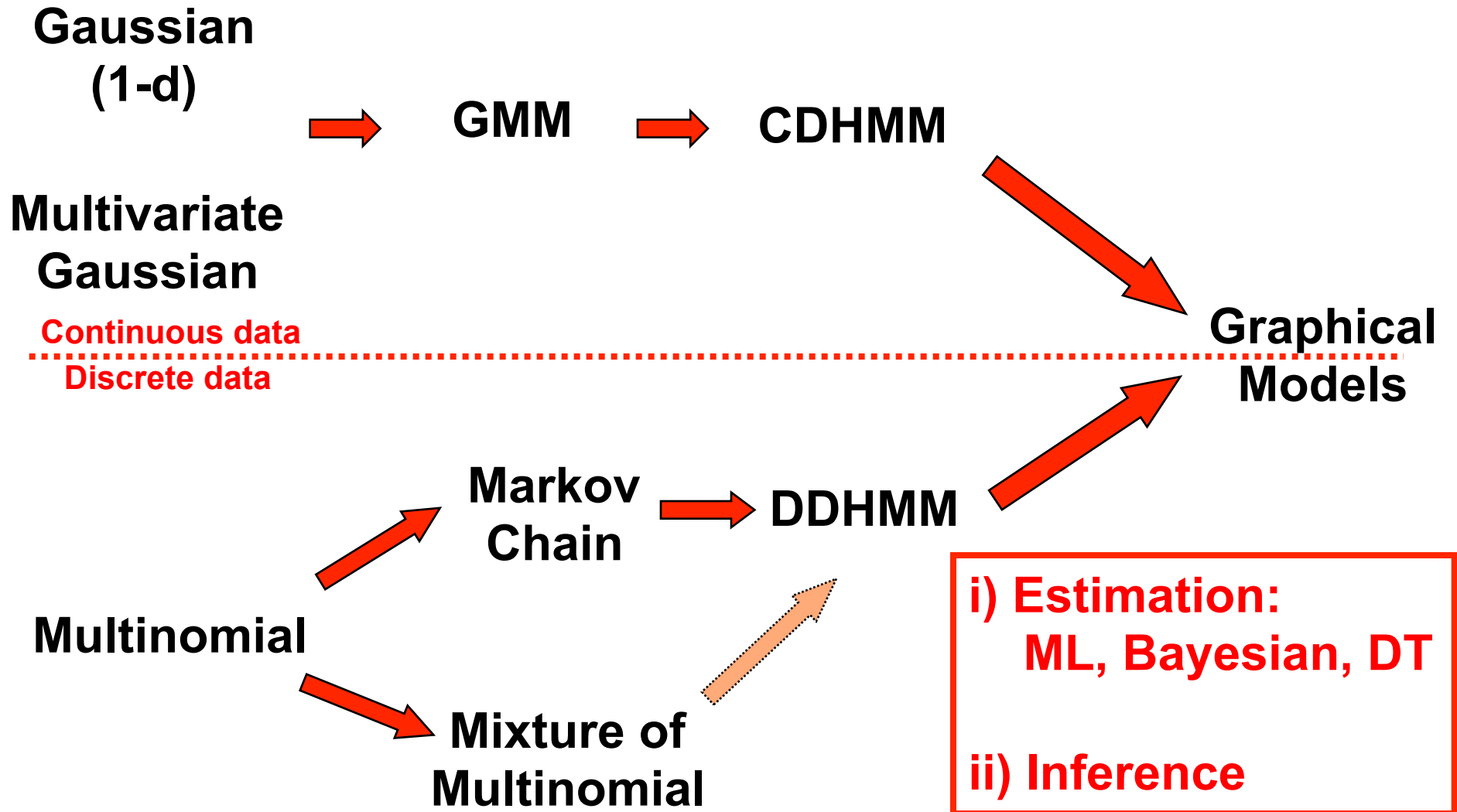


# Statistical Data Modeling

- For any real problem, the true p.d.f.'s are always unknown, neither the forms of the functions nor the parameters.
- Our approach – **statistical data modeling** : based on the available sample data set, choose a proper statistical model to fit into the available data set.
  - **Data Modeling stage**: once the statistical model is selected, its function form becomes known except the set of model parameters associated with the model are unknown to us.
  - **Learning (training) stage**: the unknown parameters can be estimated by fitting the model into the data set based on certain estimation criterion.
    - the estimated statistical model (assumed model format + estimated parameters) will give a parametric p.d.f. to approximate the real but unknown p.d.f. of each class.
  - **Decision (test) stage**: the estimated p.d.f.'s are plugged into the optimal Bayes decision rule in place of the real p.d.f.'s
    - ➔ plug-in MAP decision rule
    - Not optimal any more but performs reasonably well in practice



# Statistical Models: roadmap



# Model Parameter Estimation (1)

- **Maximum Likelihood (ML) Estimation:**
  - Objective function: likelihood function of all observed data
  - ML method: most popular model estimation; simplest
  - EM (Expected-Maximization) algorithm
  - Examples:
    - Univariate Gaussian distribution
    - Multivariate Gaussian distribution
    - Multinomial distribution
    - Gaussian Mixture model (GMM)
    - Markov chain model: n-gram for language modeling
    - Hidden Markov Model (HMM)
- **Bayesian Model Estimation**
  - The MAP (maximum *a posteriori*) estimation (point estimation)
  - General Bayesian theory for parameter estimation
  - Recursive Bayes Learning (Sequential Bayesian learning)



# Maximum Likelihood Estimation (I)

- After data modeling, we know the model form (*p.d.f.*), i.e.  $p(X | \omega_i)$ . For each class, we don't know its parameters, e.g.,  $\theta_i$ .
- To show the dependence of  $p(X | \omega_i)$  on  $\theta_i$  explicitly, we rewrite it as  $p(X | \omega_i, \theta_i)$ . We assume  $p(X | \omega_i, \theta_i)$  has a known parametric form.
- In pattern classification problem, we usually collect a sample set for each class, we have  $N$  data sets,  $D_1, D_2, \dots, D_N$ .
- The parameter estimation problem: to use the information provided by the training samples  $D_1, D_2, \dots, D_N$  to obtain good estimates for the unknown parameter vectors,  $\theta_1, \theta_2, \dots, \theta_N$ , associated with each class.



# Maximum Likelihood Estimation (II)

- The Maximum Likelihood (ML) principle: we view the parameters as quantities whose values are fixed but unknown. The best estimate of their value is defined to be the one that maximizes the probability of observing the samples actually observed.
  - Best interpret the data;
  - Fit the data best.
- The likelihood function:
  - $p(X | \theta)$  → data distribution p.d.f of different  $X$  if  $\theta$  is given
  - $p(\theta | X)$  → likelihood function of  $\theta$  if data  $X$  is given



# Maximum Likelihood Estimation (III)

- Problem: use information provided by  $D_1, D_2, \dots, D_N$  to estimate  $\theta_1, \theta_2, \dots, \theta_N$ .
- Assumption I: samples in  $D_i$  give no information about  $\theta_j$  if  $i \neq j$ . Thus we estimate parameters for each class separately and estimate each  $\theta_i$  solely based on  $D_i$ .
  - the joint estimation becomes: use a set  $D$  of training samples drawn independently from the probability density  $p(X | \theta)$  to estimate the unknown parameter vector  $\theta$ .
- Assumption II: all samples in each set  $D_i$  are *i.i.d.* (*independent and identically distributed*), i.e., the samples are drawn independently according to the same probability law  $p(X | \omega_i)$ .



# Maximum Likelihood Estimation (IV)

- Assume  $D$  contains  $n$  samples,  $X_1, X_2, \dots, X_n$ , since the samples were drawn independently from  $p(X | \theta)$ , thus the probability of observing  $D$  is

$$p(D | \theta) = \prod_{k=1}^n p(X_k | \theta)$$

- If viewed as a function of  $\theta$ ,  $p(D|\theta)$  is called the likelihood function of  $\theta$  with respect to the sample set  $D$ .
- The maximum-likelihood estimate of  $\theta$  is *the value  $\theta_{ML}$  that maximizes  $p(D|\theta)$ .*

$$\theta_{ML} = \arg \max_{\theta} p(D | \theta) = \arg \max_{\theta} \prod_{k=1}^n p(X_k | \theta)$$

- Intuitively,  $\theta_{ML}$  corresponds to the value of  $\theta$  which in some senses best agrees with or supports the actually observed training samples.





# Maximum Likelihood Estimation (V)

- In many cases, it is more convenient to work with the logarithm of the likelihood rather than the likelihood itself.
- Denote the *log-likelihood* function  $l(\theta) = \ln p(D|\theta)$ , we have

$$\theta_{ML} = \arg \max_{\theta} l(\theta) = \arg \max_{\theta} \sum_{k=1}^n \ln p(X_k | \theta)$$

- How to do maximization in ML estimation:
  - For simple models: differential calculus
    - Single univariate/multivariate Gaussian model
  - Model parameters with constraints: Lagrange optimization
    - Multinomial/ Markov Chain model
  - Complex models: Expectation-Maximization (EM) method
    - GMM/HMM



# Maximization: differential calculus

- The log-likelihood function:

$$l(\theta) \equiv \ln p(D | \theta) = \sum_{k=1}^n \ln p(X_k | \theta)$$

- Assume  $\theta$  is a  $p$ -component vector  $\theta = (\theta_1, \theta_2, \dots, \theta_p)$ , and let  $\nabla_{\theta}$  be the gradient operator as:

$$\nabla_{\theta} = \begin{bmatrix} \partial / \partial \theta_1 \\ \vdots \\ \partial / \partial \theta_p \end{bmatrix}$$

- Calculate  $\nabla_{\theta} l(\theta) = \sum_{k=1}^K \nabla_{\theta} \ln p(X_k | \theta)$

- Maximization is done by equating to zero:

$$\nabla_{\theta} l(\theta) = 0$$



# Examples of ML estimation (1): Univariate Gaussian with unknown mean

- Training data  $D = \{x_1, x_2, \dots, x_n\}$  (a set of scalar numbers)
- We decide to model the data by using a univariate Gaussian distribution, i.e.,

$$p(x | \theta) = N(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Assume we happen to know the variance, we only need to estimate the unknown mean from the data by using ML estimation.
- The log-likelihood function:

$$\begin{aligned} l(\mu) &= \ln p(D | \mu) = \ln \prod_{k=1}^n p(x_k | \mu) \\ &= \sum_{k=1}^n \ln p(x_k | \mu) = \sum_{k=1}^n \left[ -\frac{\ln(2\pi\sigma^2)}{2} - \frac{(x_k - \mu)^2}{2\sigma^2} \right] \end{aligned}$$



# Examples of ML estimation(1): univariate Gaussian with unknown mean (cont' )

- Maximization:

$$\frac{d}{d\mu} l(\mu) = 0$$

$$\Rightarrow \sum_{k=1}^n (x_k - \mu) = 0$$

$$\Rightarrow \mu_{ML} = \frac{\sum_{k=1}^n x_k}{n}$$

- ML estimate of the unknown Gaussian mean is the sample mean.



## Examples of ML estimation(2): multivariate Gaussian distribution

- Training data  $D=\{X_1, X_2, \dots, X_n\}$  (a set of vectors)
- We decide to model  $D$  with a multivariate Gaussian distribution

$$p(X | \mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left[-\frac{(X - \mu)^t \Sigma^{-1} (X - \mu)}{2}\right]$$

- Assume both mean vector and variance matrix are unknown.
- The log-likelihood function:

$$\begin{aligned} l(\mu, \Sigma) &= \ln p(D | \mu, \Sigma) = \sum_{k=1}^n \ln p(X_k | \mu, \Sigma) \\ &= C - \frac{n}{2} \ln |\Sigma| - \frac{1}{2} \cdot \sum_{k=1}^n (X_k - \mu)^t \Sigma^{-1} (X_k - \mu) \end{aligned}$$



## Examples of ML estimation(2): multivariate Gaussian distribution (Cont' )

- Maximization:

$$\frac{\partial l(\mu, \Sigma)}{\partial \mu} = 0 \Rightarrow \sum_{k=1}^n \Sigma^{-1} (X_k - \mu) = 0$$

$$\Rightarrow \mu_{ML} = \frac{1}{n} \sum_{k=1}^n X_k$$

$$\frac{\partial l(\mu, \Sigma)}{\partial \Sigma} = 0$$

$$\Rightarrow -\frac{n}{2} (\Sigma^{-1})^t + \frac{1}{2} \sum_{k=1}^n (\Sigma^{-1})^t (X_k - \mu)(X_k - \mu)^t (\Sigma^{-1})^t = 0$$

$$\Rightarrow \frac{n}{2} \Sigma = \frac{1}{2} \sum_{k=1}^n (X_k - \mu)(X_k - \mu)^t$$

$$\Rightarrow \Sigma_{ML} = \frac{1}{n} \sum_{k=1}^n (X_k - \mu)(X_k - \mu)^t$$

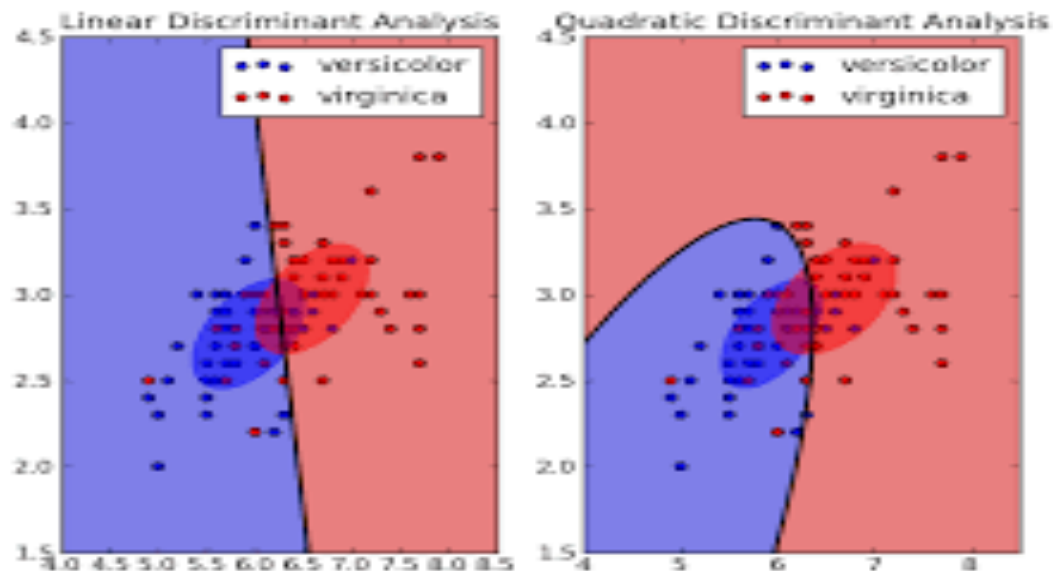


# N-class pattern classification based on Gaussian models

- Given  $N$  classes  $\{\omega_1, \omega_2, \dots, \omega_N\}$ , for each class we collect a set of training samples,  $D_i = \{X_{i1}, X_{i2}, \dots, X_{iT}\}$ , for class  $\omega_i$ .
- For each sample in the training set, we observe its feature vector  $X$  as well as its true class id  $\omega$ .
- If the feature vector is continuous and uni-modal, we may want to model each class by a multivariate Gaussian distribution,  $N(\mu, \Sigma)$ .
- Thus, we have  $N$  different multivariate distributions,  $N(\mu_i, \Sigma_i)$  ( $i=1, 2, \dots, N$ ), one for each class.
- The model forms are known but their parameters,  $\mu_i$  and  $\Sigma_i$  ( $i=1, 2, \dots, N$ ), are unknown.
- Use training data to estimate the parameters based on ML criterion.  $D_i \rightarrow \mu_i$  and  $\Sigma_i$
- Classifying any unknown pattern: when observing an unknown pattern,  $Y$ , classify with the estimated models based on the
- plug-in Bayes decision rule:  
$$\omega_Y \equiv i^* = \arg \max_i p(\omega_i) \cdot p(Y | \omega_i) = \arg \max_i N(Y | \mu_i^{ML}, \Sigma_i^{ML})$$

# Linear and Quadratic Discriminant Analysis

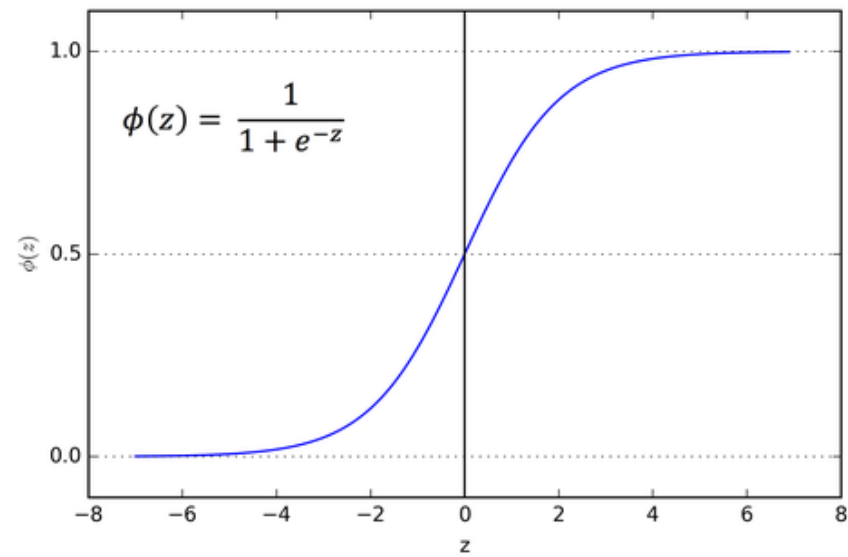
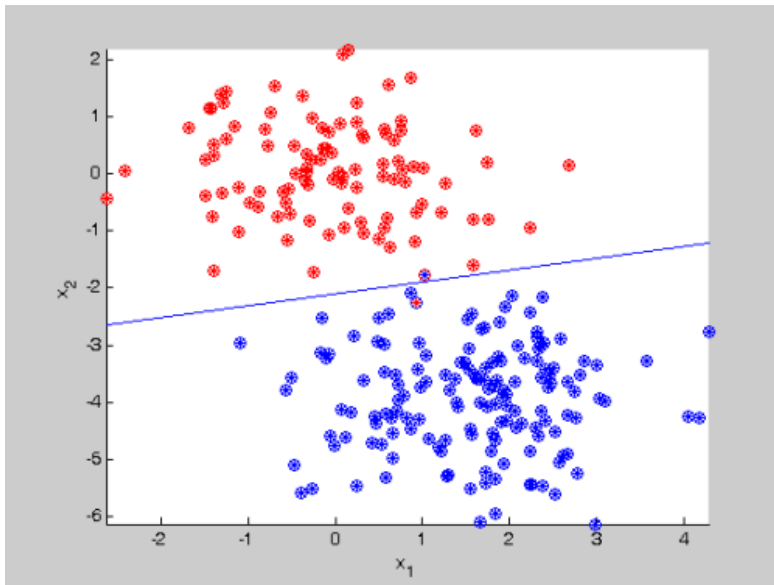
- **Pattern Classification:** each class is modeled by a multivariate Gaussian
- **Linear Discriminant Analysis**
  - Two Gaussians share the same covariance matrix
  - The decision surface is a linear hyperplane
- **Quadratic Discriminant Analysis**
  - Two Gaussians have different covariance matrices
  - The decision surface is a Quadratic hyperbola





# Examples of ML estimation(3): Logistic Regression

- Two-class pattern Classification: rely on a sigmoid function
- Probability of Class 0:  
 $\Pr(\omega=0|x) = \Phi(w \cdot x + b) = y$
- Probability of Class 1:  
 $\Pr(\omega=1|x) = 1 - \Phi(w \cdot x + b) = 1 - y$



# Examples of ML estimation(3): Logistic Regression

- Logistic Regression: model parameters  $w$  and  $b$
- Maximum Likelihood Estimation

Given a trained set  $(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots, (\mathbf{x}_N, t_N)$

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^N y_n^{t_n} \{1 - y_n\}^{1-t_n}$$

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^N \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^N (y_n - t_n) \mathbf{x}_n$$

***No closed-form solution; Requires an iterative optimization method, such as SGD, iterative reweighted least squares, etc.***

# Multi-class Logistic Regression

- For multi-class pattern classification:
- Rely on a *soft-max* function:

$$p(\omega_k|\mathbf{x}) = \frac{\exp(a_k)}{\sum_j \exp(a_j)} = \frac{\exp(\mathbf{w}_k \cdot \mathbf{x} + \mathbf{b}_k)}{\sum_j \exp(\mathbf{w}_j \cdot \mathbf{x} + \mathbf{b}_j)}$$

- Model parameters:  $\mathbf{w}_k, \mathbf{b}_k$  ( $k=1,2, \dots$ )
- Maximum Likelihood Estimation:  
 $\mathbf{x}_m^k$  :  $m$ -th sample from class  $k$

$$l(\{\mathbf{w}_k, \mathbf{b}_k\}|\mathbf{X}) = \sum_k \sum_m \ln p(\omega_k|\mathbf{x}_k^m) = \sum_k \sum_m \ln \frac{\exp(\mathbf{w}_k \cdot \mathbf{x}_k^m + \mathbf{b}_k)}{\sum_j \exp(\mathbf{w}_j \cdot \mathbf{x}_k^m + \mathbf{b}_j)}$$

$$\frac{\partial l(\cdot)}{\partial \mathbf{w}_j} = \sum_m \left[ 1 - p(\omega_j|\mathbf{x}_j^m) \right] \mathbf{x}_j^m - \sum_{k \neq j} \sum_m p(\omega_k|\mathbf{x}_k^m) \mathbf{x}_k^m$$

## Examples of ML estimation(4): multinomial distribution (I)

- A DNA sequence consists of a sequence of 4 different types of nucleotides (G, A, T, C). For example,

**X**= GAATTCTTCAAAGAGTTCCAGATATCCACAGGCAGATTCTACAAAAGAAGTGTTTCAATACTGCTCTATC  
AAAAGATGTATTCCACTCAGTTACTTTTCATGCACACATCTCAATGAAGTTCCTGAGAAAGCTTCTGTCTA  
GTTTTTATGTGAAAATATTTCCCTTTTCCATCATGGGCCTCAAAGCGCTCAAATGAACCCTTGCGAGATAC  
TAGAGAAAGACTGTTTCAAACACTGCTCTATCCAAAGAACGGTTCCTACTCTGTGAGGTGAATGCACACATC  
ACAAAGCAGTTTCTGAGAACGCTTCTGTCTAGTTTGTAGGTGAAGATATTTCCCTTTTCCCTTCATAGGCCT  
CTAATCGCTCCAAATATCCACAAGCAGATTCTTCAAATGTGTGTTTCAACACTGCTCTATCAAAGAAA  
GGTTCAAGTCTGTGAGTTGAATGCACACATCACAAGCAGTTTCTGAGAATGCCTCTGTCTAGTTTGTAT  
GTGAAGATATTTCTTTTCCGTCTTATGCCTCAAATCGCTCCAAATATCCACTTGCAGATACTTCAAAA

- If assume all nucleotides in a DNA sequence are independent, we can use multinomial distribution to model a DNA sequence,
- Use  $p_1$  to denote probability to observe G in any one location,  $p_2$  for A,  $p_3$  for T,  $p_4$  for C.
- Obviously, it meets  $\sum_{i=1}^4 p_i = 1$ . (a constraint in its parameters)
- Given a DNA sequence  $X$ , the probability to observe  $X$  is

$$\Pr(X) = C \cdot \prod_{i=1}^4 p_i^{N_i}$$



## Examples of ML estimation(4): multinomial distribution (II)

- Where  $N_1$  is frequency of G appearing in X,  $N_2$  frequency of A,  $N_3$  frequency of T,  $N_4$  frequency of C.
- Problem: estimate  $p_1, p_2, p_3, p_4$  from a training sequence X based on the maximum likelihood criterion.

- The log-likelihood function:

$$l(p_1, p_2, p_3, p_4) = \sum_{i=1}^4 N_i \cdot \ln p_i$$

- Where  $N_1$  is frequency of G in training sequence X, the similar for  $N_2, N_3$  and  $N_4$ .
- Maximization  $l(.)$  subject to the constraint  $\sum_{i=1}^4 p_i = 1$
- Use Lagrange optimization:

$$L(p_1, p_2, p_3, p_4, \lambda) = \sum_{i=1}^4 N_i \cdot \ln p_i - \lambda \left( \sum_{i=1}^4 p_i - 1 \right)$$

$$\frac{\partial}{\partial p_i} L(p_1, p_2, p_3, p_4, \lambda) = 0 \quad \Rightarrow \quad N_i / p_i - \lambda = 0$$



## Examples of ML estimation(4): multinomial distribution (III)

- Finally, we get the ML estimation for the multinomial distribution as:

$$p_i = \frac{N_i}{\sum_{i=1}^4 N_i} \quad (i = 1,2,3,4)$$

- We only need count the occurrence times (frequency) of each nucleotides in all training sequences, then the ML estimate can be easily calculated as above.
- Similar derivation also holds for Markov chain model.
  - It has an important application in language modeling, the so-called *n-gram* model.



## Examples of ML estimation(5): Markov Chain Model (I)

- **Markov assumption:** a discrete-time Markov chain is a random sequence  $x[n]$  whose  $n$ -th conditional probability function satisfy:
- $$p(x[n] | x[n-1]x[n-2]...x[n-N]) = p(x[n] | x[n-1])$$
- In other words, probability of observing  $x[n]$  only depends on its previous one  $x[n-1]$  (for 1<sup>st</sup> order Markov chain) or the most recent history (for higher order Markov chain).
- Parameters in Markov Chain model are a set of conditional probability functions.



## Examples of ML estimation(5): Markov Chain Model (II)

- **Stationary assumption:**

$$p(x[n] | x[n-1]) = p(x[n'] | x[n'-1]) \text{ for all } n \text{ and } n'.$$

- **For stationary discrete Markov Chain model:**
  - Only one set of conditional probability function
- **Discrete observation:** in practice, the range of values taken on by each  $x[n]$  is finite, which is called state space. Each distinct one is a Markov state.
  - An observation of a discrete Markov chain model becomes a sequence of Markov states.
  - The set of conditional probs  $\rightarrow$  transition matrix





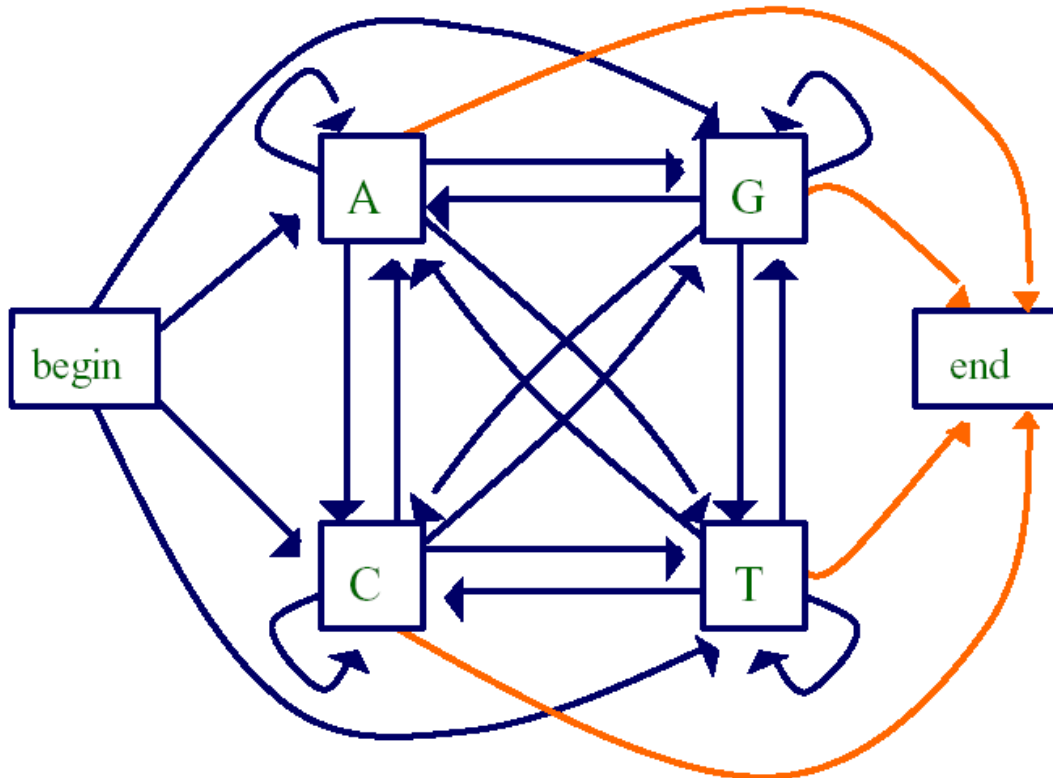
# Examples of ML estimation(5): Markov Chain Model (III)

- Markov Chain Model (stationary & discrete):
  - A finite set of Markov states, to say  $M$  states.
  - A set of state conditional probabilities, i.e., *transition matrix*  
In 1<sup>st</sup> order Markov chain model,  $a_{ij} = p(j|i) \quad (i,j=1,2,\dots,M)$
- Markov Chain model can be represented by a directed graph.
  - Node  $\rightarrow$  Markov state
  - Arc  $\rightarrow$  state transition (each arc attached with a transition probability)
  - A Markov chain observation can be viewed as a path traversing a Markov chain model.
- Probability of observing a Markov chain can be calculated based on the path and the transition matrix.



# Examples of ML estimation(5): Markov Chain Model (IV)

- First-order Markov Chain Model for DNA sequence



Full Transition matrix (6 by 6)

$$p(A|G) = 0.16$$

$$p(C|G) = 0.34$$

$$p(G|G) = 0.38$$

$$p(T|G) = 0.12$$

...

...

One transition probability is attached with each arc.

$$Pr(GAATTC) = p(\text{begin})p(G|\text{begin})p(A|G)p(A|A)p(T|A)p(T|T)p(C|T)p(\text{end}|C)$$

# Examples of ML estimation(5): Markov Chain Model (V)

- Markov Chain Model for language modeling (*n*-gram)
  - Each word is a Markov state, total *N* words (vocabulary size)
  - A set of state (word) conditional probabilities

- Given any a sentence:

**S** = I would like to fly from New York to Toronto this Friday

- 1<sup>st</sup>-order Markov chain model: ***N*\**N*** conditional probabilities

$$Pr(\mathbf{S}) = p(I|begin) p(would|I) p(like|would) p(to|like) p(fly|to) \dots$$

- This is called bi-gram model

- 2<sup>nd</sup>-order Markov chain model: ***N*\**N*\**N***

$$Pr(\mathbf{S}) = p(I|begin) p(would|I,begin) p(like|would,I) p(to|like,would) p(fly|to,like) \dots$$

- This is called tri-gram model

- Multinomial (0<sup>th</sup>-order Markov chain): ***N*** probabilities

$$Pr(\mathbf{S}) = p(I) p(would) p(like) p(to) p(fly) \dots$$

- This is called uni-gram model



## Examples of ML estimation(5): Markov Chain Model (VI)

- How to estimate Markov Chain Model from training data
  - Similar to ML estimate of multinomial distribution
  - Maximization of log-likelihood function with constraints.

- Results:

$$p(W_i | W_j) = \frac{\text{Frequency of } W_j W_i \text{ in training data}}{\text{Frequency of } W_j \text{ in training data}}$$

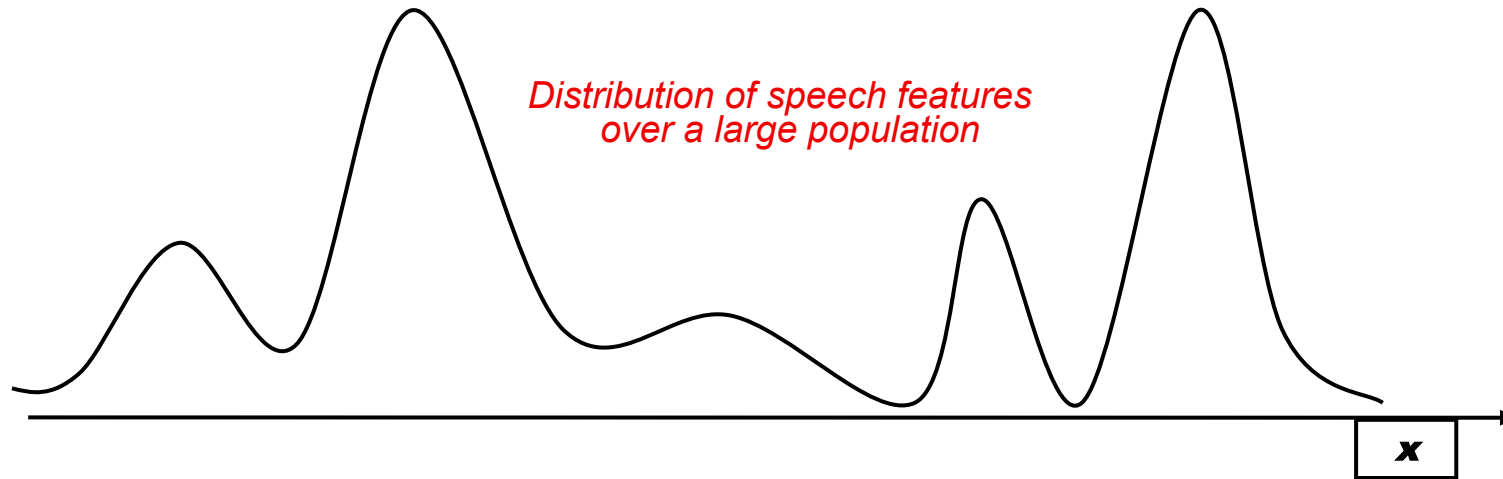
$$p(W_i | W_j, W_k) = \frac{\text{Frequency of } W_k W_j W_i \text{ in training data}}{\text{Frequency of } W_k W_j \text{ in training data}}$$

- Generally, N-gram model: a large number of probabilities to be estimated.



# Examples of ML estimation(6): Gaussian Mixture Model (GMM) (I)

- Single Gaussian distribution (either univariate or multivariate) is a single mode distribution.
- In many cases, the true distribution of data is complicated and has multiple modes in nature.



- For this kind of applications, better to use a more flexible model
  - Gaussian Mixture model (GMM)
  - A GMM can be tuned to approximate any arbitrary distribution



# Examples of ML estimation(6): Gaussian Mixture Model (GMM) (II)

- **Gaussian Mixture model (GMM)**

- **Univariate density**

$$p(x) = \sum_{k=1}^K \omega_k \cdot N(x | \mu_k, \sigma_k^2)$$

- **Multivariate density**

$$p(X) = \sum_{k=1}^K \omega_k \cdot N(X | \mu_k, \Sigma_k)$$

- **GMM is a mixture of single Gaussian distribution (each one is called mixand) which have different means and variances.**

- **$\omega_k$  is called mixture weight, prior probability of each mixand.**

They satisfy  $\sum_{k=1}^K \omega_k = 1$  .

- **GMM is widely used for speaker recognition, audio classification, audio segmentation, etc.**



# Examples of ML estimation(6): Gaussian Mixture Model (GMM) (III)

- However, estimation of a GMM is not trivial.
- Consider a simple case:
  - We have a set of training data  $D=\{x_1, x_2, \dots, x_n\}$
  - Use a 2-mixture GMM to model it:

$$p(x) = \frac{0.3}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} + \frac{0.7}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}$$

- We try to get ML estimate of  $\mu_1, \sigma_1, \mu_2, \sigma_2$  from training data.
- Simple maximization based on differential calculus does not work.
  - For each  $x_i$ , we don't know which mixture it comes from. The number of terms in likelihood function  $p(D | \mu_1, \sigma_1, \mu_2, \sigma_2)$  increases exponentially as we observe more and more data.
  - No simple solution.
- Need alternative method – Expectation Maximization (EM) algorithm



# The Expectation-Maximization (EM) Algorithm (I)

- EM algorithm is an iterative method of obtaining maximum likelihood estimate of model parameters.
- EM suits best to the so-called *missing data* problem:
  - Only observe a subset of features, called *observed*,  $X$ .
  - Other features are *missing* or unobserved, denoted as  $Y$ .
  - The complete data  $Z=\{X, Y\}$ .
  - If given the complete data  $Z$ , it is usually easy to obtain ML estimation of model parameters.
  - How to do ML estimation based on observed  $X$  only??





# Expectation-Maximization (EM) Algorithm (II)

- Initialization: find an initial values for unknown parameters
- EM algorithm consists of two steps:
  - Expectation (E-step): the expectation is calculated with respect of the missing data  $Y$ , using the current estimate of the unknown parameters and conditioned upon the observed  $X$ .
  - Maximization (M-step): provides a new estimate of unknown parameters (better than the initial ones) in terms of maximizing the above expectation  $\rightarrow$  increasing likelihood function of observed.
- Iterate until convergence



# EM Algorithm(1): E-step

- E-step: form an auxiliary function

$$Q(\theta; \theta^{(i)}) = E_Y \left[ \ln p(X, Y | \theta) \mid X, \theta^{(i)} \right]$$

- The expectation of log-likelihood function of complete data is calculated based on the current estimate of unknown parameter, and conditioned on the observed data.
- $Q(\theta; \theta^{(i)})$  is a function of  $\theta$  with  $\theta^{(i)}$  assumed to be fixed.
- If missing data  $Y$  is continuous:

$$Q(\theta; \theta^{(i)}) = \int_{\Lambda_Y} \ln p(X, Y | \theta) \cdot p(Y | X, \theta^{(i)}) dY$$

- If missing data  $Y$  is discrete:

$$Q(\theta; \theta^{(i)}) = \sum_Y \ln p(X, Y | \theta) \cdot p(Y | X, \theta^{(i)})$$



## EM Algorithm(2): M-step

- **M-step: choose a new estimate  $\theta^{(i+1)}$  which maximizes  $Q(\theta; \theta^{(i)})$**

$$\theta^{(i+1)} = \arg \max_{\theta} Q(\theta; \theta^{(i)})$$

- $\theta^{(i+1)}$  is a better estimate in terms of increasing likelihood value  $p(X|\theta)$  than  $\theta^{(i)}$

$$p(X | \theta^{(i+1)}) \geq p(X | \theta^{(i)})$$

- **Replace  $\theta^{(i+1)}$  with  $\theta^{(i)}$  and iterate until convergence.**



# EM Algorithm(3)

- EM algorithm guarantees that the log-likelihood of the observed data  $p(X|\theta)$  will increase monotonically.
- EM algorithm may converge to a local maximum or global maximum. And convergence rate is reasonably good.
- Applications of the EM algorithm:
  - ML estimation of some complicated models, e.g., GMM, HMM, ... (in general mixture models of e-family)
  - ET (emission tomography) image reconstruction
  - Active Noise Cancellation (ANC)
  - Spread-spectrum multi-user communication



# An Application of EM algorithm: ML estimation of multivariate GMM(I)

- Assume we observe a data set  $D=\{X_1, X_2, \dots, X_T\}$  (a set of vectors)
- We decide to model the data by using multivariate GMM:

$$p(X) = \sum_{k=1}^K \omega_k \cdot N(X | \mu_k, \Sigma_k) \quad \left(\text{with } \sum_{k=1}^K \omega_k = 1\right)$$

- Problem: use data set D to estimate GMM model parameters, including  $\omega_k, \mu_k, \Sigma_k$  ( $k=1, 2, \dots, K$ ).
- If we know the label of mixand  $l_t$  from which each data  $X_t$  come from, the estimation is easy.
- Since the mixand label is not available in training set, we treat it as missing data:
  - Observed data:  $D=\{X_1, X_2, \dots, X_T\}$ .
  - Missing data:  $L=\{l_1, l_2, \dots, l_T\}$ .
  - Complete data:  $\{D, L\} = \{X_1, l_1, X_2, l_2, \dots, X_T, l_T\}$



# An Application of EM algorithm: MLE of multivariate GMM(II)

- **E-step:**

$$\begin{aligned} Q(\{\omega_k, \mu_k, \Sigma_k\} | \{\omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)}\}) &= \sum_L \ln p(D, L | \{\omega_k, \mu_k, \Sigma_k\}) \cdot p(L | D, \{\omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)}\}) \\ &= C + \sum_L \left[ \sum_{t=1}^T [\ln \omega_{l_t} - \frac{n}{2} \ln |\Sigma_{l_t}| - \frac{1}{2} \cdot (X_t - \mu_{l_t})^t \Sigma_{l_t}^{-1} (X_t - \mu_{l_t})] \cdot \prod_{t=1}^t p(l_t | X_t, \{\omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)}\}) \right] \\ &= C + \sum_{k=1}^K \sum_{t=1}^T [\ln \omega_k - \frac{n}{2} \ln |\Sigma_k| - \frac{1}{2} \cdot (X_t - \mu_k)^t \Sigma_k^{-1} (X_t - \mu_k)] \cdot p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)}) \end{aligned}$$



# An Application of EM algorithm: MLE of multivariate GMM(III)

- **M-Step:**

$$\frac{\partial Q}{\partial \mu_k} = 0 \Rightarrow \mu_k^{(i+1)} = \frac{\sum_{t=1}^T X_t \cdot p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{t=1}^T p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})}$$

$$\frac{\partial Q}{\partial \Sigma_k} = 0 \Rightarrow \Sigma_k^{(i+1)} = \frac{\sum_{t=1}^T (X_t - \mu_k^{(i)})^t \cdot (X_t - \mu_k^{(i)}) \cdot p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{t=1}^T p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})}$$

$$\frac{\partial}{\partial \omega_k} [Q - \lambda (\sum_{k=1}^K \omega_k - 1)] = 0 \Rightarrow \omega_k = \frac{\sum_{t=1}^T p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{k=1}^K \sum_{t=1}^T p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})} = \frac{\sum_{t=1}^T p(l_t = k | X_t, \omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)})}{T}$$



# An Application of EM algorithm: MLE of multivariate GMM(IV)

- where

$$p(l_t = k | X_t, \{\omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)}\}) = \frac{\omega_k^{(i)} \cdot N(X_t | \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{k=1}^K \omega_k^{(i)} \cdot N(X_t | \mu_k^{(i)}, \Sigma_k^{(i)})}$$

- Iterative ML estimation of GMM
  - Initiation: choose  $\{\omega_k^{(0)}, \mu_k^{(0)}, \Sigma_k^{(0)}\}$ . Usually use vector clustering algorithm (such as K-means) to cluster all data into K clusters. Each cluster is used to train for one Gaussian mix.
  - $i=0$ ;
  - Use EM algorithm to refine model estimation
$$\{\omega_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)}\} \Rightarrow \{\omega_k^{(i+1)}, \mu_k^{(i+1)}, \Sigma_k^{(i+1)}\}$$
  - $i++$ , go back until convergence.





# **GMM Initialization: K-Means clustering**

- **K-Means Clustering: a.k.a. unsupervised learning**
- **Unsupervisedly cluster a data set into many homogeneous groups**
- **K-Means algorithm:**
  - **step 1: assign all data into one group; calculate centroid.**
  - **step 2: choose a group and split.**
  - **step 3: re-assign all data to groups.**
  - **step 4: calculate centroids for all groups.**
  - **step 5: go back to step 3 until convergence.**
  - **step 6: stop until K classes**
- **Basics for clustering:**
  - **distance measure**
  - **centroid calculation**
  - **choose a group and split**



# Applications of GMM

- **GMM is widely used to model speech or audio signals. In many cases, we use diagonal covariance matrices in GMM for simplicity.**
- **Speaker recognition:**
  - **Collect some speech signals from all known speakers.**
  - **Train a GMM for each known speaker by using his/her voice.**
  - **For an unknown speaker, prompt him/her to say sth.**
  - **Classify it based on all trained GMM's and determine the speaker's identity or reject.**
- **Audio classification:**
  - **Classify a continuous audio/video stream (from radio or TV) into some homogeneous segments: anchor's speech, in-field interview, telephone interview, music, commercial ads, sports, etc.**
  - **For each category, train a GMM based on training data.**
  - **Use all trained GMM's to scan an unknown audio stream to segment it.**



# Bayesian Theory

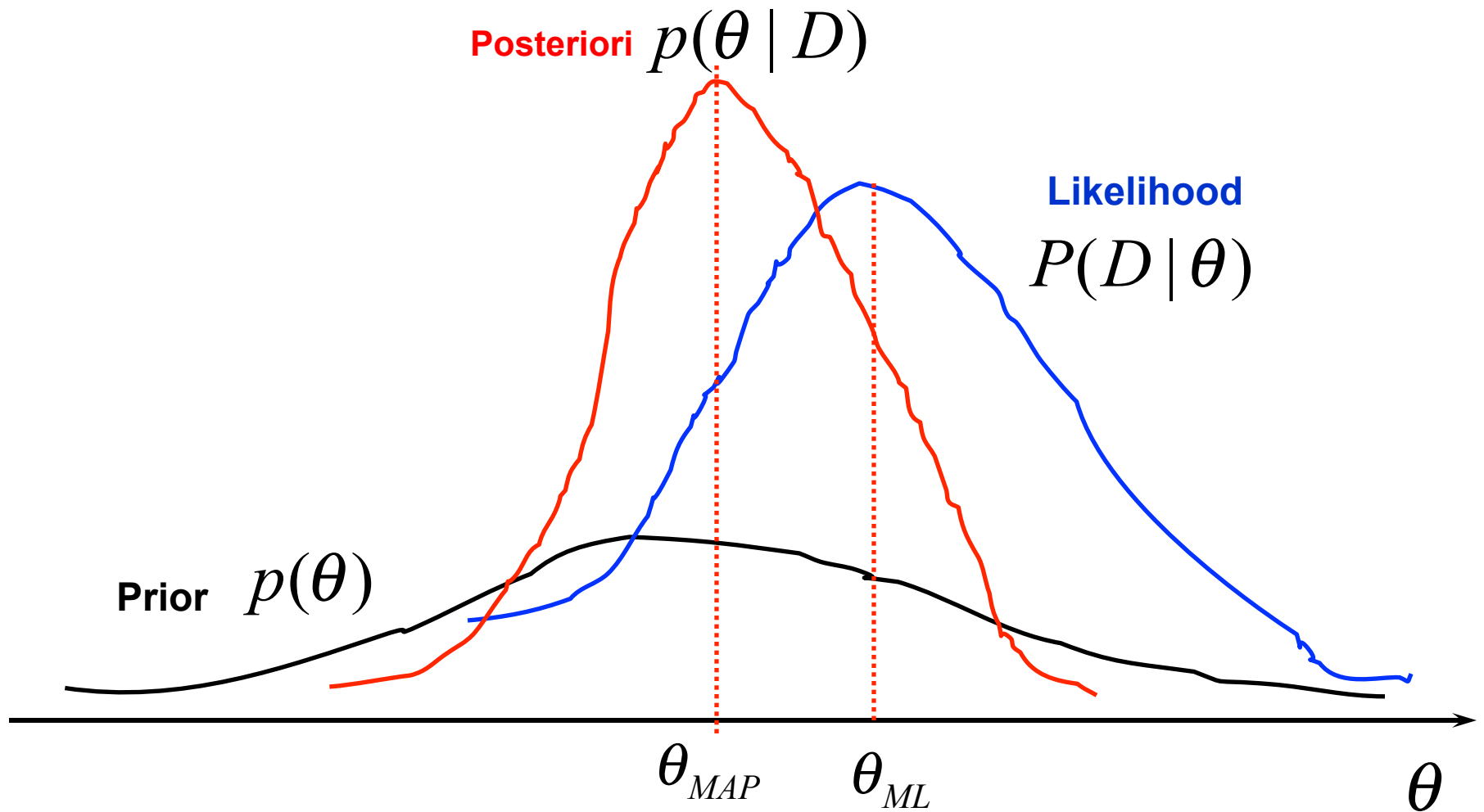
- Bayesian methods view model parameters as random variables having some known prior distribution. **(Prior specification)**
  - Specify prior distribution of model parameters  $\theta$  as  $p(\theta)$ .
- Training data  $D$  allow us to convert the prior distribution into a posteriori distribution. **(Bayesian learning)**

$$p(\theta | D) = \frac{p(\theta) \cdot p(D | \theta)}{p(D)} \propto p(\theta) \cdot p(D | \theta)$$

- We infer or decide everything solely based on the posteriori distribution. **(Bayesian inference)**
  - Model estimation: the MAP (maximum a posteriori) estimation
  - Pattern Classification: Bayesian classification
  - Sequential (on-line, incremental) learning
  - Others: prediction, model selection, etc.



# Bayesian Learning



# The MAP estimation of model parameters

- Do a point estimate about  $\theta$  based on the posteriori distribution

$$\theta_{MAP} = \arg \max_{\theta} p(\theta | D) = \arg \max_{\theta} p(\theta) \cdot p(D | \theta)$$

- Then  $\theta_{MAP}$  is treated as estimate of model parameters (just like ML estimate). Sometimes need the EM algorithm to derive it.
- MAP estimation optimally combine prior knowledge with new information provided by data.
- MAP estimation is used in speech recognition to adapt speech models to a particular speaker to cope with various accents
  - From a generic speaker-independent speech model → prior
  - Collect a small set of data from a particular speaker
  - The MAP estimate give a speaker-adaptive model which suit better to this particular speaker.

# Bayesian Classification

- Assume we have  $N$  classes,  $\omega_i$  ( $i=1,2,\dots,N$ ), each class has a class-conditional pdf  $p(X|\omega_i,\theta_i)$  with parameters  $\theta_i$ .
- The prior knowledge about  $\theta_i$  is included in a prior  $p(\theta_i)$ .
- For each class  $\omega_i$ , we have a training data set  $D_i$ .
- Problem: classify an unknown data  $Y$  into one of the classes.
- The Bayesian classification is done as:

$$\omega_Y = \arg \max_i p(Y | D_i) = \arg \max_i \int p(Y | \omega_i, \theta_i) \cdot p(\theta_i | D_i) d\theta_i$$

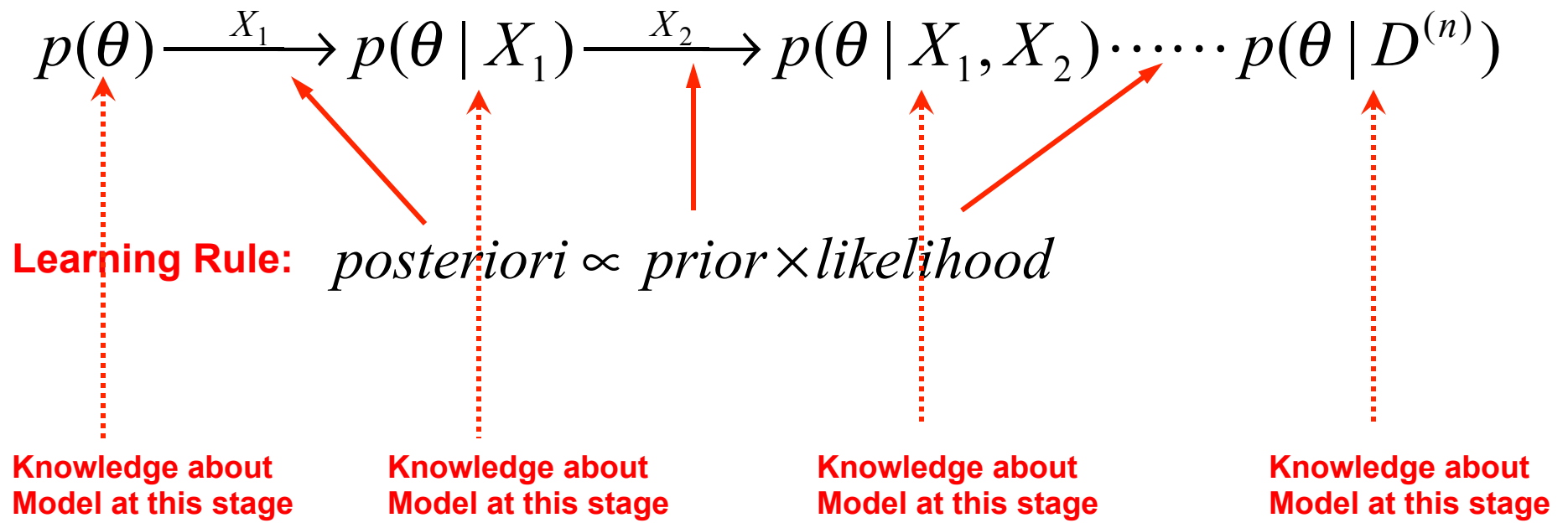
where

$$p(\theta_i | D_i) = \frac{p(\theta_i) \cdot p(D_i | \omega_i, \theta_i)}{p(D_i)} \propto p(\theta_i) \cdot p(D_i | \omega_i, \theta_i)$$



# Recursive Bayes Learning (On-line Bayesian Learning)

- Bayesian theory provides a framework for *on-line learning* (a.k.a. *incremental learning*, *adaptive learning*).
- When we observe training data one by one, we can dynamically adjust the model to learn incrementally from data.
- Assume we observe training data set  $D=\{X_1, X_2, \dots, X_n\}$  one by one,



# How to specify priors

- **Noninformative priors**
  - In case we don't have enough prior knowledge, just use a flat prior at the beginning.
- ***Conjugate priors*: for computation convenience**
  - For some models, if their probability functions are a reproducing density, we can choose the prior as a special form (called *conjugate prior*), so that after Bayesian learning the posterior will have the exact same function form as the prior except the all parameters are updated.
  - Not every model has conjugate prior.





# Conjugate Prior

- For a univariate Gaussian model with only unknown mean:

$$p(x | \omega_i) = N(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x - \mu)^2}{2\sigma^2}\right]$$

- If we choose the prior as a Gaussian distribution (Gaussian's conjugate prior is Gaussian)

$$p(\mu) = N(\mu | \mu_0, \sigma_0^2) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left[-\frac{(\mu - \mu_0)^2}{2\sigma_0^2}\right]$$

- After observing a new data  $x_1$ , the posterior will still be Gaussian:

$$p(\mu | x_1) = N(\mu | \mu_1, \sigma_1^2) = \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp\left[-\frac{(\mu - \mu_1)^2}{2\sigma_1^2}\right]$$

where

$$\mu_1 = \frac{\sigma_0^2}{\sigma_0^2 + \sigma^2} x_1 + \frac{\sigma^2}{\sigma_0^2 + \sigma^2} \mu_0$$

$$\sigma_1^2 = \frac{\sigma_0^2 \sigma^2}{\sigma_0^2 + \sigma^2}$$



# The sequential MAP Estimate of Gaussian

- For univariate Gaussian with unknown mean, the MAP estimate of its mean after observing  $x_1$ :

$$\mu_1 = \frac{\sigma_0^2}{\sigma_0^2 + \sigma^2} x_1 + \frac{\sigma^2}{\sigma_0^2 + \sigma^2} \mu_0$$

- After observing next data  $x_2$ :

$$\mu_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma^2} x_2 + \frac{\sigma^2}{\sigma_1^2 + \sigma^2} \mu_1$$

